TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky
a mezioborových studií

VYUŽITÍ VÝVOJOVÝCH DESEK LPCEXPRESSO V MOBILNÍCH ROBOTECH

Bakalářská práce

Studijní program: B2646 – Informační technologie
Studijní obor: 1802R007 – Informační technologie

Autor práce: **Jakub Hirnšal**
Vedoucí práce: Ing. Miroslav Holada, Ph.D.

Liberec 2015

TECHNICAL UNIVERSITY OF LIBEREC
Faculty of Mechatronics, Informatics
and Interdisciplinary Studies

USE LPCEXPRESSO DEVELOPMENT BOARDS IN MOBILE ROBOTS

Bachelor thesis

Study programme: B2646 – Information Technology
Study branch: 1802R007 – Information Technology

Author: **Jakub Hirnšal**
Supervisor: Ing. Miroslav Holada, Ph.D.

Liberec 2015

TECHNICKÁ UNIVERZITA V LIBERCI
Fakulta mechatroniky, informatiky a mezioborových studií
Akademický rok: 2014/2015

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Jakub Hirnšal**
Osobní číslo:	**M12000332**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Informační technologie**
Název tématu:	**Využití vývojových desek LPCExpresso v mobilních robotech**
Zadávající katedra:	**Ústav informačních technologií a elektroniky**

Zásady pro vypracování:

1. Seznamte se s vývojovými deskami LCExpresso od firmy NXP, které jsou osazeny mikrokontroléry řady LPC1xxx. Zaměřte se na komunikaci s dalšími obvody a ovládání digitálních vstupů a výstupů.
2. Ověřte digitální vstupy a výstupy mikrokontroléru a možnosti připojení vybraných obvodů (např. snímač teploty, vnější paměť nebo MEMS senzor).
3. Navrhněte ukázkovou aplikaci, která bude demonstrovat využití mikrokontroléru v mobilním robotu.
4. Navrženou aplikaci realizujte a zhodnoťte zjištěné výhody a nevýhody implementace dané vývojové desky s mikrokontrolérem oproti známým stavajícím realizacím na pracovišti školitele.

Rozsah grafických prací: **Dle potřeby dokumentace**

Rozsah pracovní zprávy: **cca 30 stran**

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

[1] **Novák, Petr. Mobilní roboty - pohony, senzory, řízení. Praha : BEN - technická literatura, 200str. 24ISBN 80-7300-141-1.**

Vedoucí bakalářské práce: **Ing. Miroslav Holada, Ph.D.**
Ústav informačních technologií a elektroniky

Datum zadání bakalářské práce: **12. září 2014**

Termín odevzdání bakalářské práce: **15. května 2015**

prof. Ing. Václav Kopecký, CSc.
děkan

L.S.

prof. Ing. Zdeněk Plíva, Ph.D.
vedoucí ústavu

V Liberci dne 12. září 2014

Prohlášení

Byl jsem seznámen s tím, že na mou bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb., o právu autorském, zejména § 60 – školní dílo.

Beru na vědomí, že Technická univerzita v Liberci (TUL) nezasahuje do mých autorských práv užitím mé bakalářské práce pro vnitřní potřebu TUL.

Užiji-li bakalářskou práci nebo poskytnu-li licenci k jejímu využití, jsem si vědom povinnosti informovat o této skutečnosti TUL; v tomto případě má TUL právo ode mne požadovat úhradu nákladů, které vynaložila na vytvoření díla, až do jejich skutečné výše.

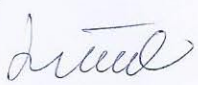
Bakalářskou práci jsem vypracoval samostatně s použitím uvedené literatury a na základě konzultací s vedoucím mé bakalářské práce a konzultantem.

Současně čestně prohlašuji, že tištěná verze práce se shoduje s elektronickou verzí, vloženou do IS STAG.

Datum:

15.5.2015

Podpis:



Poděkování

Rád bych touto cestou poděkoval panu **Ing. Miroslavu Holadovi, Ph.D.** za možnost pracovat na katedře i bez jakýchkoliv předchozích znalostí z tohoto oboru. Také bych mu chtěl poděkovat za vstřícný přístup při řešení problémů a dotazů.

Anotace

Česky:

Úkolem této bakalářské práce je seznámení se s vývojovými deskami LPCExpresso od firmy NXP, které jsou osazeny mikrokontroléry řady LPC1xxx. Zjistit možnosti komunikace s dalšími obvody a ovládání digitálních vstupů a výstupů. Tyto možnosti ověřit na vybraných obvodech.

Práce je tedy rozdělena do několika částí. V první části je seznámení s mikrokontroléry LPC1xxx. V druhé části se práce zaměřuje na komunikaci vybraných obvodů, které jsou pak následně připojeny a otestovány. Dále je navržena ukázková aplikace, která demonstruje využití mikrokontroléru s vybranými obvody na mobilním robotu. V poslední části je na navržená aplikace realizována, otestována a zhodnocena.

Klíčová slova:

Mikrokontrolér, NXP, LPCExpresso, LPC1xxx, LPC1769, MEMS2 senzor, BTM 110, I2C, UART, FreeRTOS.

Anotace

English:

The goal of this work is to get familiar with development board LPCExpresso created by NXP, which are equipped with a series of microcontrollers LPC1xxx. Identifying possibilities for communication with other control circuitry and digital inputs/outputs. These posibilities check on selected circuits.

The thesis is divided into several parts. The first part is an familiarization with microcontrollers LPC1xxx. The second part is focused on communication with selected circuits that are subsequently connected and tested. Design a application, that demonstrates the use of a microcontroller with selected circuits on a mobile robot is the next part. In the last part, designed application is implemented, tested and evaluated.

Keywords:

Microcontroller, NXP, LPCExpresso, LPC1xxx, LPC1769, MEMS2 senzor, BTM 110, I2C, UART, FreeRTOS.

Zkratky a názvy

PWM Pulse Width Modulation

ADC Analog to Digital Converter

DAC Digital to Analog Converter

I2C Inter-Integrated Circuit

SCL Synchronous Clock

SDA Synchronous Data

USB Universal Serial Bus

PCI-X Peripheral Component Interconnect Extended

LCD A liquid-crystal display

COM Component Object Model

Vcc power-supply pin

Vss ground pin

NFC Near field communication

JTAG Joint Test Action Group

Obsah

Seznam obrázků a tabulek

1 Úvod

Používání chytrých a mobilních zařízení je v dnešní době naprosto běžným jevem. Dnes lze již bez problému v jakýchkoliv elektrických zařízeních nalézt řídící jednotku. Své místo mají nejen v systémech, kde je potřeba složité řízení nebo ovládání, ale používají se rovněž k řízení různých částí automobilu včetně motoru, v přenosných lékařských přístrojích nebo implantátech, dálkových ovládáních, kancelářských přístrojích, měřících přístrojích, hračkách, mp3 přehrávačích, mobilních telefonech a v celé řadě dalších přístrojů a zařízení. Čipy v zařízeních jsou stále rychlejší, úspornější a zvládají stále více výpočetně náročnější úlohy. Mezi hlavní výrobce čipů patří: Atmel, Freescale, NXP, STMicroelectronics a další. Výhodou použití mikrokontroléru je, že dokáže nahradit velké množství logických obvodů a diskrétních součástek, které byly dříve k realizaci podobných zapojení potřeba. Díky různým periferiím, které jsou již v mikrokontroléru integrovány, dokáže nahradit i další integrované obvody. Pomocí zvyšujícímu se výkonu čipů je možné po vyhodnocení vstupů a dat, integrovat složité výpočty a pomocí komunikačních kanálů posílat pouze konečné data, či ovládat další obvody.

Motivací této práce je zjištění možností využití výkonných čipů pro inteligentní roboty. Cílem práce je realizovat ukázkovou úlohu, která demonstruje tyto možnosti využití čipů v praxi. Firma NXP nabízí celou řadu vývojových desek s výkonnými ARM čipy. Pro tuto bakalářskou práci byla vybrána vývojová deska z řady LPC1xxx. Výhodou těchto desek je, že obsahují i ladící část včetně JTAGu. Stačí je zapojit pomocí mini-USB konektoru s počítačem a stáhnout vývojové prostředí LPCXpresso IDE.

První část bakalářské práce seznamuje s vývojovými deskami LPCExpresso a mikrokontroléry řady LPC1xxx. Dále se také zaměřuje na komunikaci s dalšími obvody a ovládání digitálních vstupů a výstupů. Následně se ověřují možnosti připojení vybraných obvodů jako je MEMS2 senzor. V další části je navržena ukázková aplikace, která demonstruje využití mikrokontroléru v mobilním robotu. Navržená aplikace je následně realizována. Ve výsledku byly zjištěné výhody a nevýhody implementace dané vývojové desky s mikrokontrolérem.

Další motivací této práce je shrnout zjištěné možnosti a zkušenosti pro další použití, protože zatím není nikde dostupné.

2 Produkty firmy NXP

Firma NXP Semiconductors patří mezi největší výrobce čipů na světě. Sídlí ve městě Eindhoven v Nizozemí. Byla divizí nizozemského výrobce elektroniky Philips, v samostatný podnik se změnila v roce 2006. Dnes působí ve více než 25 zemí světa. Kromě výroby mikrokontrolerů se firma zaměřuje na řadu dalších produktů, jako jsou zesilovače, rádia, senzory, NFC, čtečky. Vše připojené, chytré a bezpečné. Z toho vyplívá i jejich moto „Secure Connections for a Smarter World.“ NXP nabízí přes 500 variant mikrokontrolerů LPCExpresso. Tato bakalářská práce se zaměřuje mikrokontrolery LPCExpresso řady LPC1xxx.

2.1 LPC1xxx

Mikrokontrolery řady LPC1xxx jsou založené na 32-bitových Cortex jádrech z architektury procesorů ARM, pracují rychlostí až 180MHz. Řady LPC11xx a LPC12xx jsou založeny na jádře Cortex-M0, série LPC13xx, 17xx a18xx na Cortex-M3. Pro pochopení rozdílů těchto čipů je třeba objasnit pojmy Cortex-M0 a M3.

2.1.1 Cortex-M0

Cortex-M0 je 32-bitový ARM čip z architektury ARMv6-M. Jedná se o nejúspornější a nejlevnější čip z řady Cortex-M.

- 50 MHz/45 DMips
- 100 μA/MHz
- THUMB instrukční sada bez: CBZ, CBNZ, IT
- THUMB-2 pouze : BL, DMB, DSB, ISB, MRS, MSR
- 1-32 přerušení a NMI

2.1.2 Cortex-M3

Cortex-M3 je první čip z řady Cortex-M. Je vytvořen na architektuře ARMv7-M. Tento čip nabízí velmi mnoho periferních připojení a aplikací.

- 180 MHz/135 DMips
- Až 200 μA/MHz pro LPC13xx
- THUMB (celá)
- THUMB-2 (celá)
- 1 – 240 přerušení a NMI
- Integrovaný sleep mode

3 Vývojové prostředí

LPCXpresso IDE je integrované vývojové prostředí pro mikrokontroléry LPC do firmy NXP. Je navrženo pro jednoduché použití. Obsahuje všechny nástroje potřebné pro vývoj, které umožnují vytvořit kvalitní kód za krátkou dobu s malými náklady. Základem prostředí je vývojové prostředí Eclipse obsahující mnoho rozšíření, která mají zjednodušit vývoj s mikrokontrolery LPC. Obsahuje též průmyslové nástroje GNU s možností C knihoven nebo standardních Newlib knihoven. Vývojové prostředí má 2 licence: Free a Pro. Ve free licenci je možné vytvořit plně optimalizovaný kód do velikosti 256kB. Pro verze nemá žádné omezení a poskytuje jeden rok e-mailové podpory přímo od NXP techniky. Obě licence se po instalaci vývojového prostředí musí aktivovat. Free licence vyžaduje registraci vývojového prostředí na oficiálních stránkách NXP. Pro licenci je třeba koupit a poté aktivovat ve vývojovém prostředí.

3.1 Instalace

Instalace není nijak složitá. Instalační soubor lze stáhnout přímo z oficiálních stránek: www.lpcware.com. Po stažení a spuštění instalačního programu je možno vývojové prostředí nainstalovat do libovolného adresáře. Lze nainstalovat i více verzí prostředí bez jakýchkoliv problémů.

3.1.1 Požadavky na systém

Výrobcem testované operační systémy jsou:

- Microsoft® Windows - XP 32-bit (SP2 nebo novější)
- Microsoft® Windows - Vista 32-bit nebo 64-bit
- Microsoft® Windows - Windows 7 32-bit nebo 64-bit
- Microsoft® Windows – Windows 8
- Mac OS X 10.7.5 (Lion), and 10.8.2 (Mountain Lion)
- Linux - Ubuntu 9 trough 12Linux - Fedora 14 and 17

Minimální paměť RAM je 2 GB MB, doporučená je 4 GB. 500+ MB volného místa na disku. Je doporučený rychlý internet pro stažení a registrování vývojového prostředí.

LPCXpresso IDE je možná naistalovat a spustit i na jiných linuxových distribucích. Nicméně, testovány byly pouze výše zmíněné distribuce. Nástroje pro virtualizaci linuxu či windows s USB podporou lze také použít ke spuštění LPCXpresso pro jiné platformy.

3.2 Aktivace

Aktivace LPCXpresso IDE free edice není nijak složitá. Ve vývojovým prostředí je třeba v záložce Help - > Activate zvolit možnost „Create seriál number and register.“ Poté je třeba zkopírovat či opsat sériové číslo LPCXpresso, které je založeno hardwaru a konfiguraci operačního systému a nemělo by obsahovat osobní údaje. Dále je třeba přejít na registrační stránku pomocí nabízeného odkazu. Zde se zobrazí registrační formulář. Po vyplnění údajů a sériového čísla LPCXpresso bude poslán email s aktivačním kódem. Nyní stačí kód vložit do vývojového prostředí v záložce Help -> Activate -> Activate (Free edition).

V případě Pro licence, je třeba aktivační kód zakoupit na internetovém obchodu NXP. V dalším kroku nutno zvolit v LPCXpresso IDE možnost „Activate(Pro edition)“ v záložce Help -> Activate. Po vložení aktivačního kódu se musí zadat validní emailová adresa a libovolné heslo, které bude asociované s aktivačním kódem. Toto heslo lze dále využít například pro deaktivaci Pro verze vývojového prostředí či k jeho reaktivaci.

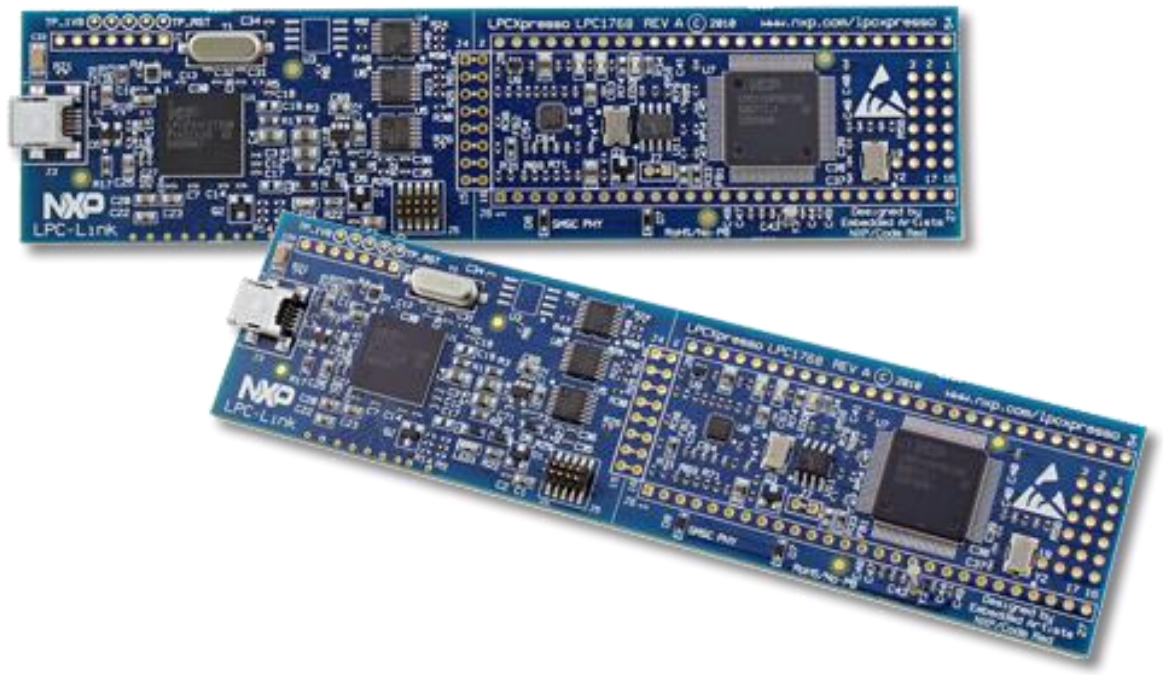
4 Vývojová deska LPC1769

Pro tuto bakalářskou práci byla zvolena vývojová deska LPCExpresso LPC1769. Tato deska obsahuje výše zmíněný čip ARM Cortex-M3, který představuje vysokou úroveň integrace a nízkou spotřebu. LPC1769 pracuje na frekvencích CPU až do 120 MHz. Cortex – M3 obsahuje tří úrovňovou pipeline (rouru) a je založen na Harvardské architektuře s oddělenými lokálními instrukcemi a datovými sběrnicemi, stejně tak, jako třetí sběrnici pro periferní zařízení. LPC1769 obsahuje 512KB flash pamět pro program a 64KB paměť pro data.

Významné periferní doplňky pro LPC1769:

- **Ethernetová MAC adresa**
- **USB device interface –** komunikace typu Master–Slave, komunikace s počítačem
- **USB OTG interface** – On-The-Go, je přidaná specifikace do klasického USB standardu, přináší možnost přímého přenosu bod-bod(point-to-point), jedná se o komunikaci dvou periferních zařízení bez potřeby intervence počítače.
- **USB host interface** – hostitelské zařízení, master
- **8-channel DMA controller** – umožnuje přístup do hlavní systémové paměti nezávisle na CPU pro některé hardwarové subsystémy
- **4x UART**
- **2x CAN**
- **2x SSP kontrolér**
- **SPI**
- **3x I2C**
- **I2S 2x vstup a 2x vystup**
- **8-kanálový 12-bitový ADC**
- **10-bitový DAC**
- **PWM na ovládaní motoru**
- **6x PWM –** pro všeobecné použití
- **RTC** - s vnějším napájením s baterie
- **4x časovač/čítač**
- **70x I/O**

LPC1769 má 32KB SRAM se sběrnicí pro lepší přístup do CPU. Dále 2x 16KB SRAM bloky s odděleným přístupem pro vyšší propustnost. Tyto bloky paměti mohou být použity pro Ethernet, USB, DMA paměti a samozřejmě také pro CPU instrukce a pro data.



Obrázek 1 – LPC1769
zdroj: http://www.embeddedartists.com/sites/default/files/image/product/xpr_lpc176x_banner.png

5 Operační systém reálného času

FreeRTOS je operační systém reálného času pro vestavěné zařízení. Tento systém je celý napsán v programovacím jazyce C a pouze několik funkcí je vytvořeno v assembleru. Jádro je složeno ze tří souborů a to list.c, queue.c a task.c. FreeRTOS je jednou ze tří distribucí výrobce, která je zdarma. Další 2 verze jsou OpenRTOS (komerční použití) a dále je to SafeRTOS, který splňuje bezpečností normu SIL3. Pro tuto bakalářskou práci byla zvolena distribuce FreeRTOS, protože se jedná o verzi, která je zdarma a podporuje LPC1769 a procesor Cortex-M3. Tato distribuce podporuje také velké množství platforem: ARM (ARM7, ARM9 , Cortex-M4, Cortex-A), Atmel AVR, AVR32, HCS12, MicroBlaze, Cortus (APS1, APS3, APS3R, APS5, FPF3, FPS6, FPS8).

Operační systém reálného času (RTOS) je takový systém, u kterého správnost výsledku nezáleží pouze na logickém výsledku, ale také na čase, za který je výsledku dosaženo. Systém se dělí na soft a hard. Případě hard RTOS se úloha musí stihnout do stanoveného časového limitu. Nestihnutí může mít katastrofální následky. Příklad může být ABS systém v automobilu. Na rozdíl od hard real time OS se u soft real time OS dovolují drobné odchylky v reakcích. RTOS se využívá k vytvoření aplikace takové, která je napsána jako sada nezávislých vláken. Pro napsání dobré aplikace není RTOS důležitý, je to možnost volby programátora. Ale použití RTOS přináší řadu výhod jako je: Multitasking, plánovač a kontextové přepínání. RTOS je založen na přepínání vláken. Každému vláknu je přiřazena určitá priorita. Čím menší číslo tím je menší priorita.

Výše zmíněný multitasking je způsob, jak umožnit běh více nezávislých procesů na jednom procesoru. Základem operačního systému je jádro. Jádro, jako je například v operačním systému Linux, umožňuje uživatelům přistupovat k počítači zdánlivě současně. Každý prováděný proces je vlákno pod kontrolou operačního systému. Systém, který umožňuje multitasking, je schopen provádět několik procesů současně.

Na následujícím obrázku je znázorněna definice vlákna napsaná v jazyce C ve FreeRTOS. Vlákno se stará o blikání LED diody s periodou 0.5s.

```
189
190 /* LED1 toggle thread */
191 static void vLEDTask1(void *pvParameters) {
192     bool LedState = false;
193
194     while (1) {
195         Board_LED_Set(0, LedState);
196         LedState = (bool) !LedState;
197         vTaskDelay(500);
198     }
199 }
200
```

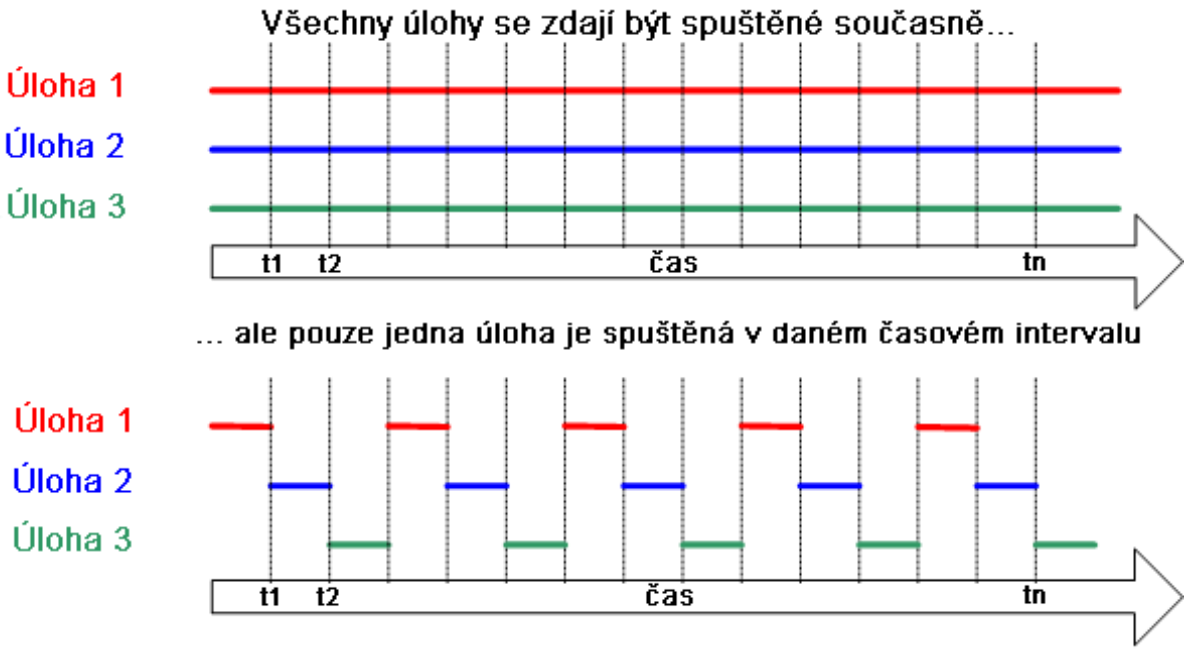
Obrázek 2 – Ukázka vlákna

Na dalším obrázku je vidět vytvoření vlákna a uvedení do FreeRTOS systému. Dalším příkazem se pak spustí všechny vytvořené vlákna. Systém se postará o jejich spouštění a přepínání.

```
520
521     /* LED1 toggle thread */
522     xTaskCreate(vLEDTask1, (signed char *) "vTaskLed1",
523                 configMINIMAL_STACK_SIZE, NULL, (tskIDLE_PRIORITY + 1UL),
524                 (xTaskHandle *) NULL);
525
526
```

Obrázek 3 Vytvoření vlákna

Použití multitasking operačního systému může zjednodušit návrh systému, co by jinak mohla být složitá aplikace. Multitasking a mezi-vláknová komunikace umožňují složité aplikaci, aby mohla být rozdělena na několik menších a více přehlednějších vláken. Rozdělení ve výsledku přinese jednodušší testování, opětovné použití kódu a například jednodušší práci v týmech. Složité načasování a sekvenční detaily mohou být odstraněny z kódu aplikace a stanou se povinností operačního systému. Běžné procesory mohou vykonávat pouze jeden úkol současně. Rychlé přepínání mezi vlákny se jeví, jako kdyby byly prováděny současně. Průběh je znázorněn na obrázku (4). Obrázek znázorňuje tři vlákna, která běží současně. V horní polovině obrázku je zobrazen vnímaný průběh chodu vláken a v dolní části obrázku je skutečný model multitaskingu.



Obrázek 4 Princip spouštění úloh

Další výhodnou funkcí RTOS je plánovač. Plánovač je funkce operačního systému, která postupně zpracovává běžící vlákna. Pro představu: plánovač si vytvoří seznam vláken, která jsou připravena k běhu. Když neběží nějaký proces, tedy je procesor volný, tak si plánovač vybere jedno vlákno z vytvořeného seznamu a přiměje vlákno ke spuštění. V preemptivním plánování jsou vlákna pozastavována v určitých periodách přerušení. Plánovač vybere nové vlákno, a jakmile se vrátí z přerušení, tak bude spuštěno nové vlákno. V tomto případě si systém sám určuje, kdy bude vlákno pozastaveno a kdy se vrátí do běžícího stavu. V kooperativním plánovači se vlákna samy rozhodují, kdy se vrátí z pozastaveného stavu do běžícího. Distribuce FreeRTOS, která je používána v projektu, podporuje preemptivní a také kooperativní plánování [5].

Poslední důležitá funkce je kontextové přepínání. Když vlákno běží, využívá registrů RAM a ROM paměti mikrokontroléru, stejně jako jakýkoliv jiný program. Tyto zdroje dohromady (registry procesoru a stack) tvoří kontext běžícího vlákna. Vlákno je část kódu, které neví, kdy bude pozastaveno nebo obnoveno jádrem. Ale také neví, kdy se to stane. Například vlákno je pozastaveno těsně předtím, než běží instrukce, která sečte 2 hodnoty registrů. Když je vlákno pozastaveno, ostatní vlákna budou běžet a můžou změnit hodnoty registrů procesoru. Po obnovení vlákno nebude vědět, že byly hodnoty registru změněny a následkem toho by byla špatná hodnota výsledku. Aby bylo zabráněno tomuto typu chyby, je

nezbytné, aby po obnovení vlákna mělo vlákno identický kontext, jako těsně před jeho pozastavením. Jádro operačního systému je zodpovědné za to, že uloží kontext vlákna, když je pozastaveno. Jakmile je opět obnoveno, tak jádro operačního systému obnoví uložený kontext těsně předtím, než je vlákno spuštěno. Proces ukládání kontextu vlákna, které je pozastaveno a obnovování kontextu vlákna, které je obnoveno, se nazývá kontextové přepínání.

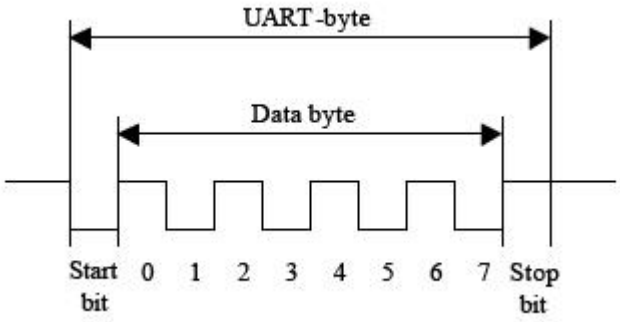
6 Komunikace UART

Zkratka UART (Universal Asynchronous Receiver and Transmitter) lze přeložit jako univerzální asynchronní přijímač a vysílač. Jedná se o část hardwaru, která pomocí dvou pinů označovaných jako RX a TX přijímá a odesílá data. Jelikož se jedná o asynchronní způsob komunikace, musí přijímač a vysílač obsahovat k řízení komunikace vlastní generátor hodinového signálu. Je potřeba nastavit rychlost hodin, velikost jednoho bajtu, počet stop bitů a paritní bit. Často můžeme také slyšet zkratku USART. Jedná se v podstatě o to samé, pouze s tím rozdílem, že je USART obsahuje i synchronní režim.

USART může být spuštěn v těchto režimech:

- **Asynchronní (full duplex)** – plně duplexní asynchronní systém, například pro komunikaci s periferiemi jako jsou CRT terminály, osobní počítače (PC) atd.
- **Synchronní (half duplex)** – „poloduplexní“ synchronní systém, A/D a D/A převodníky, sériová EEPROM atd. Lze nastavit jako - Master nebo Slave.

UART vysílá data na pinu TX (transmit) a přijímá na pinu RX (receive). Pokud neprobíhá příjem ani vysílání, je signál nastaven na log. 1 (klidový režim). Vysílání je zahájeno Start-bitem, při kterém se změní hodnota signálu na log. 0 po dobu jednoho bitu. Data se posílají od nejnižšího bitu. Poslední nejvýznamnější datový bit je následován stop bitem, který má úroveň log. 1 stejně jako klidový režim. Po odvysílání stop-bitu může začít přenos dalšího bajtu. Celý proces je znázorněn na následujícím obrázku (5).



Obrázek 5 – UART byte, zdroj: http://automatizace.hw.cz/obrazek/lin_uart.jpg

6.1 LPC1769 a USART

LPC1769 jako i jiné vývojové desky a čipy obsahují několik UART registrů. Komunikace mezi procesorem a UART je kompletně pod kontrolou dvanácti z těchto registrů. Z registrů lze číst nebo do nich zapisovat pro kontrolu nebo změnu chování komunikačního zařízení. Každý registr má osm bitů. LPC1769 jsou všechny registry uloženy ve struktuře a je jich celkem 26. Některé se používají pouze pro USART. Nejdůležitější registry jsou popsány níže.

RBR : Receiver buffer register (RO)

Tento registr obsahuje jeden příchozí bajt, pokud není použit žádný zasobník, nebo poslední nepřečtený bajt ze zásobníků. Pokud je použit zásobník, kazdé čtení z tohoto registru vrátí následující bajt, dokud nějaký bude k dispozici. Bit 0 v LSR(Line status register) se využívá ke kontrole, zda jsou všechny bajty již přečtené. Pokud již není co číst, bit 0 v LSR se nastaví na hodnotu 0.

THR : Transmitter holding register (WO)

THR se používá „bufferování“ odchozích znaků. Pokud není použit žádný zásobník, lze do THR uložit pouze jeden znak. Bit 5 v LSR značí, zda je třeba zapsat další znak do THR. Hodnota 1 v LSR znamená, že registr je prázdný. Pokud je použit nějaký zásobník, je možné zapsat další znak. THR se nepoužívá přímo pro posílání dat na TX. Bajt je přenesen do dalšího registru, kde se data posílají na TX bit po bitu (shift register).

IER : Interrupt enable register (R/W)

Nejlepším způsobem komunikace je řízení přes přerušení. UART signalizuje veškeré změny pomocí procesorového přerušení, které je softwarově ošetřeno. Přerušení se zapínají nastavením bitů v IER registru do log. 1.

- Bit 0 : zapnutí přerušení po přijetí dat
- Bit 1: zapnutí signalizace prázdného THR
- Bit 2: RLS přerušení

- Bit 3: Modem status přerušení
- Bit 4: zapnutí sleep mode
- Bit 5: zapnutí low power mode

IIR : Interrupt identification register (RO)

UART vyvolá procesorové přerušení, které je třeba ošetřit. K tomu jsou nutné vědět informace o stavu UARTU. Tento stav je uložen v IIR registru.

LCR : Line control register (R/W)

LCR se používá při inicializaci pro nastavení komunikačních parametrů. Registr také řídí přístup k DLL a DLM registrů.

Bit	Hodnota			popis
0,1	Bit 0	Bit 1		Delka slova
	0	0		5bitů
	0	1		6
	1	0		7
	1	1		8
2	0			1stop bit
	1			1,5stop bit(5bit)
3,4,5	Bit 5	Bit 4	Bit 3	
	x	x	0	Žádna parita
	0	0	1	Lichá
	0	1	1	Sudá
	1	0	1	Vysoká parita
	1	1	1	Nizká parita
6	0			Zapnutí break sig.
	1			vypnutí break sig
7	0			DLAB : RBR, THR and IER přístup
	1			DLAB : DLL and DLM prístup

Tabulka 1 - LCR

FCR : FIFO control register (WO)

Tento registr řídí chování zásobníků v UARTu.

MCR : Modem control register (R/W)

MCR slouží k synchronizaci se zařízením. Umožnuje přímou manipulaci všech čtyřech pinů na UARTU.

LSR : Line status register (RO)

LSR ukazuje aktuální stav komunikace. Jsou zde zobrazeny chyby i stavy zásobníku.

Bit	Popis
0	Data k dispozici
1	Overrun error
2	Parity error
3	Framing errror
4	Obdržen break signal
5	Prázdné THR
6	Prázdné THR a linka v nečinnosti
7	Chybné údaje v zásobníku

Tabulka 2 – LSR

MSR : Modem status register (RO)

MSR obsahuje informace o stavu modemu.

SCR : Scratch register (R/W)

SCR registr nemá významné funkce. Slouží pro uložení jednobajtové informace.

DLL and DLM : Divisor latch registers (R/W)

Tyto registry slouží k ovládání rychlosti komunikace. Pro generování rychlosti používá každý UART oscilátor, který generuje frekvenci okolo 1.8432 MHz. Tato frekvence vydělena číslem 16 pro dosažení základní (maximální) rychlosti 115200 bps. Pokud je požadována menší rychlost, je tato základní rychlost dále dělena. Pokud je dělitel větší než 255, je uložen ve dvou oddělených registrech. V DLL je jsou uloženy nižší bity dělitele a v DLM vyšší.

Pro nastavení parametrů UART komunikace v LPC1769 (i pro ostatní desky) jsou připravené funkce, definice a makra v hlavičkových souborech. Stačí si pouze přečíst, co daná funkce řeší a jaké přejímá parametry. Pak stačí metodu zavolat a předat ji správné parametry.

Na následujícím obrázku (6) je znázorněna ukázková konfigurace UARTu. UART_SELECTION je makro, ve kterém je uloženo hexadecimální číslo aktivního UARTu. V další parametr je 19200, což je stanovená rychlost. Zbylé parametry jsou makra ve kterých jsou definovány nastavení, které uloží do příslušných registrů. Makra jsou definována v hlavičkové souboru uart_17xx_40xx.h.

```
105         Chip_UART_Init(UART_SELECTION);
106         Chip_UART_SetBaud(UART_SELECTION, 19200);
107         Chip_UART_ConfigData(UART_SELECTION, (UART_LCR_WLEN8 | UART_LCR_SBS_1BIT));
108         Chip_UART_SetupFIFOS(UART_SELECTION, (UART_FCR_FIFO_EN | UART_FCR_TRG_LEV2));
109         Chip_UART_TXEnable(UART_SELECTION);
```

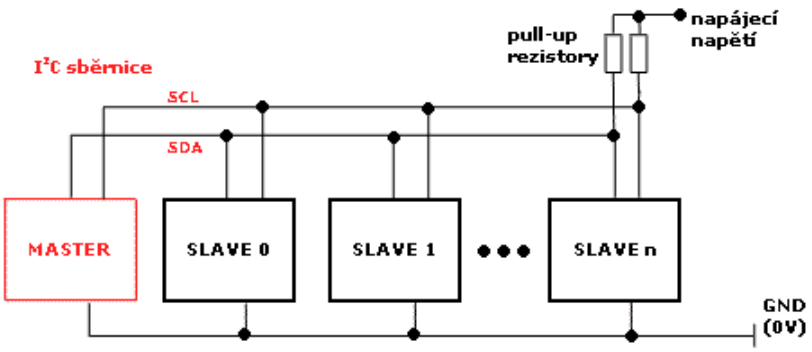
Obrázek 6 – Konfigurace UART v LPC1769

- UART_LCR_WLEN8 : délka slova 8 bitů
- UART_LCR_SBS_1BIT : nastavení pro 1 stop bit.
- UART_FCR_FIFO_EN : zapnutí zásobníku
- UART_FCR_TRG_LEV2 : nastavení trigger level2 pro zásobník, 8 znaků

Pro posílání či příjem dat je napsáno hned několik funkcí. Jsou funkce pro posílání jednoho bajtu či celé pole. Stejné to je i u přijímání dat. Pro tyto funkce je třeba vytvořit nějaký zásobník, např. kruhový buffer.

7 I2C komunikace

I²C bus je zkratka, která vznikla z IIC bus, tedy Internal-Integrated-Circuit Bus. Jak již název napovídá, jedná se o interní datovou sběrnici sloužící pro komunikaci a přenos dat mezi jednotlivými integrovanými obvody většinou v rámci jednoho zařízení. Vyvinula ji firma Philips přibližně před 20 lety a od té doby prošla několika vylepšeními, ale o tom až později. V dnešní době tento způsob "komunikace" podporuje řada integrovaných obvodů nejen firmy Philips. Jedná se především o mikrokontroléry, sériové paměti, inteligentní LCD, audio, video obvody, a/d a d/a převodníky a některé další digitálně řízené obvody. Hlavní výhodou je, že obousměrný přenos probíhá pouze po dvou vodičích - "data SDA (serial data)" a "hodiny SCL (serial clock)". To především u mikrokontrolérů výrazně optimalizuje nároky na počet vstupně-výstupních pinů a celkově zjednodušuje výsledné zapojení. Na jednu sběrnici může být připojeno více integrovaných obvodů. V základní verzi jsou obvody adresovány 7bitově a v rozšířené verzi 10bitově. To umožňuje připojení 128 respektive 1024 čipů s různou adresou na jednu společnou sběrnici. V praxi jsou tato čísla však podstatně nižší, protože adresa čipu většinou nelze určit plnými 7 (10) bity ale třeba jen třemi. Někdy nelze určit vůbec a je dána na pevno pro daný typ čipu - takových čipů tedy na jedné sběrnici nemůže být více než jeden. Přenosová rychlost sběrnice je pro většinu aplikací dostatečná i v základní verzi, kde je frekvence hodin 100kHz. Ve vylepšených verzích to může být 400kHz nebo 1MHz, ale ne všechny integrované obvody tuto verzi podporují. Rychlost přenosu pak musí být přizpůsobena pochopitelně "nejpomalejšímu" čipu na sběrnici. Oba vodiče musí být implicitně v logické jedničce a to je zajištěno pull-up rezistory. Jejich odpory mají hodnotu v řádech jednotek kiloohmů. Čím je vyšší komunikační frekvence, tím musí být nižší hodnoty těchto odporů. Pro 100kHz postačuje 4k7. [4]



Obrázek 7 - Zapojení I2C sběrnice

Jeden z integrovaných obvodů (většinou mikrokontrolér) je nastaven jako MASTER a všechny ostatní obvody jsou SLAVE. Obvody se dají zapojit i jako tzv. multi-master, kdy je čipů master několik. V tomto článku se však omezím pouze na zapojení s jedním Master čipem a 7 bitovou adresací. [4]

Master při jakémkoli přenosu generuje hodinový signál na vodiči SCL. Když jeden čip vysílá, přijímají všechny ostatní a pouze podle adresy určují, zda jsou data určena jim. Čip, který chce vyslat/přijmout data musí nejprve definovat adresu čipu, s kterým chce komunikovat a zda půjde o příjem nebo vysílání - tedy o čtení nebo zápis. To určuje R/W (read/write) bit, který je součástí adresy. [4]

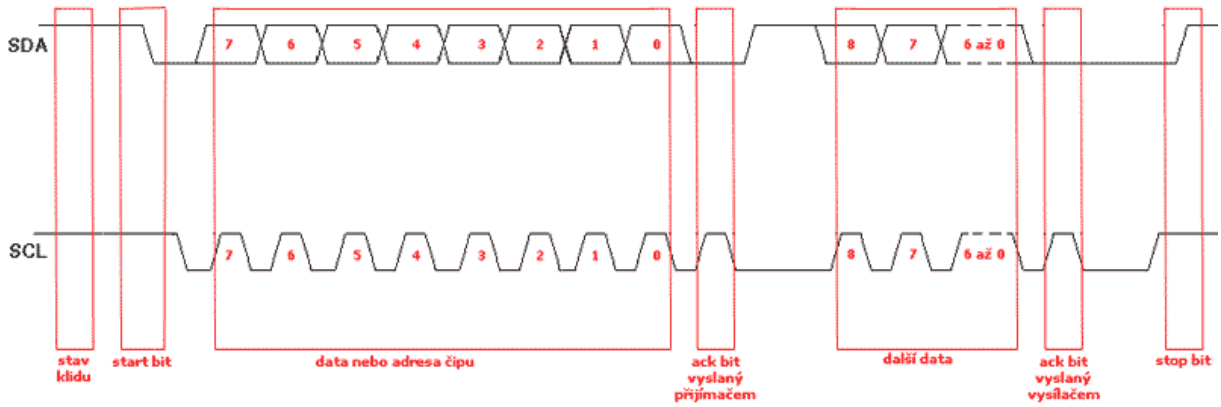
Klidový stav – logické jedničky na obou vodičích, které jsou zajištěny pull-up rezistory (mezi vodičem a napájecím napětím). Klidový stav nastane, pokud jsou výstupy obvodu master ve stavu vysoké impedance (tedy odpojeny).

start bit - Zahajuje přenos nebo jeho další část. Je vygenerován tak, že se změní úroveň SDA z 1 na 0 zatímco je SCL v logické 1.

stop bit - Ukončuje přenos. Je vygenerován podobně jako start bit. Logická úroveň SDA se změní z 0 na 1 zatímco je SCL v logické 1. Stop bit může být generován pouze po "nepotvrzení přenosu", tedy pouze po přijmutí Ack v logické 1. (viz níže)

přenos dat - Data jsou přenášena po 1Byte tedy 8 po sobě jdoucích bitů od nejvyššího po nejnižší. Při přenosu dat se může logická úroveň na SDA měnit pouze, pokud je SCL v logické 0. Při každém pulzu na SCL je přenesen jeden bit.

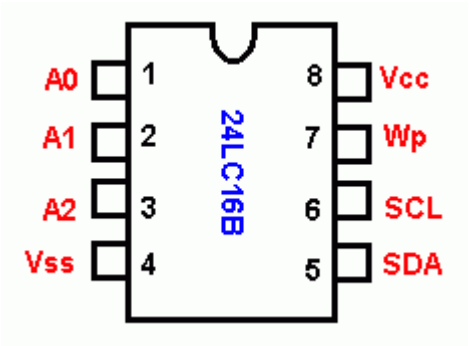
potvrzující bit Ack (acknowledge) - Tento bit slouží k potvrzení správného přijmutí dat. Ack bit se odesílá stejným způsobem jako by se odesílal devátý bit dat, ale s tím rozdílem, že ho generuje čip, který přijímal (přijímač) a nikoliv ten který data odesílal. Pokud přenos proběhl v pořádku tak odešle logickou 0. Logická 0 potvrzujícího bitu znamená rovněž to, že je přijímač připraven na příjem dalšího byte, který následuje okamžitě po něm při dalším pulzu na SCL. Pokud přenos selhal odešle logickou 1. Nebo pokud má dojít k ukončení přenosu, tak "neodešle nic". Pull-up rezistor pak zajistí, že bude na SDA logická 1 a Ack bit (v logické 0) odešle vysílač.[4]



Obrázek 8 - Časový průběh logických úrovní na vodičích SDA a SCL

8 Paměť 24LC16B

Tuto paměť vyrábí firma Microchip technology. Jedná se o paměť typu EEPROM (Eraseable Programable Read OnlyMemory). Paměť je organizovaná jako 8 bloků po 256 bajtech tj. 2048 bajtů. Na sběrnici se tedy chová jako 8 zařízení. Proto nemůžeme pracovat s vice jak jednou pamětí na jedné lince. Paměť pracuje na napětí od 2.5V do 5.5V avšak v dokumentaci je psáno až do 7V. Je kompatibilní s 100kHz (při 2.5V) a 400kHz (při 5V). Tyto frekvence se na v I2c definují jako I2cslow (100kHz) a i2cfast(400kHz) při 4MHz mikrokontroléru. 2 ms typický zapisovací cyklus pro page-write. Vydrží více než 1 000 000 zápisů nebo mazaní a data v paměti vydrží uchována více než 200let. Kompatibilní s I2C. Pin Wp - 7 slouží jako ochrana dat před přepsáním nebo znehodnocením. Pokud pin přivedeme do stavu high, paměť nelze přepisovat. Naopak stav low zaručuje přepis dat. Proto je dobré připojit pin do low, i když s ním nebudeme manipulovat, pro zaručení přepisu dat.



Obrázek 9 - 24LC16B schéma

8.1.1 Bloky paměti

Jak už bylo zmíněno, paměť 24LC16b je rozdělena do osmi bloků, které mají svá čísla adress. Nejnižší bit je tedy „vynechán“ a píše se do něj 0.

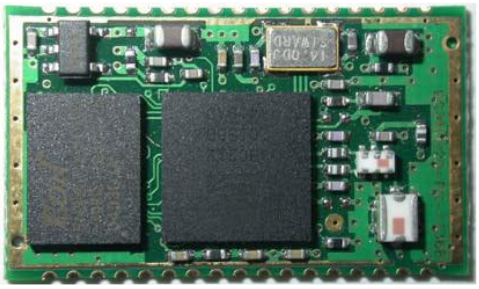
I2C slave addresy pro jednotlivé bloky v paměti:

- Block 0 Slave Address = %10100000 = 160
- Block 1 Slave Address = %10100010 = 162

- Block 2 Slave Address = %10100100 = 164
- Block 3 Slave Address = %10100110 = 166
- Block 4 Slave Address = %10101000 = 168
- Block 5 Slave Address = %10101010 = 170
- Block 6 Slave Address = %10101100 = 172
- Block 7 Slave Address = %10101110 = 174

9 Bezdrátový modul BTM 110

Jedná se o Bluetooth modul třídy 2. Maximální vysílací výkon je až 4 dBm. Tento modul je založen na CSR BlueCore 4, čipové sady, která podporuje Bluetooth V2.1 + EDR standard. Díky SPP profilu může modul komunikovat maximální přenosovou rychlost až 921.6Kbps po UART rozhraní. Vedle tohoto profilu je také modul vhodný i pro aplikace pracující s daty, jako jsou DUN, OPP, FTP, PBAP a HID. Hlasové aplikace HSP a HFP jsou k dispozici za pomoci externích hlasových kodeků PCM přes sběrnici. Modul obsahuje i USB rozhraní, které může být použito jako převodník z USB na bezdrátový UART.



Obrázek 10 – BTM110

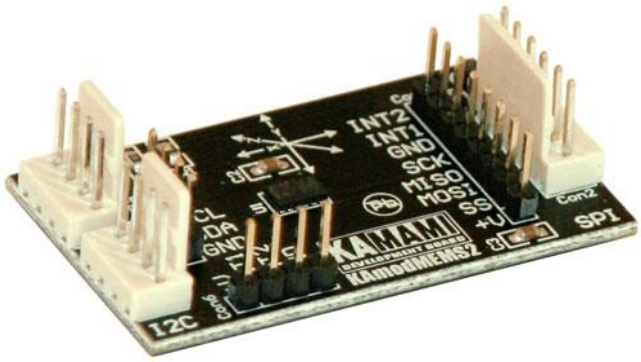
Vlastnosti:

- vnitřní 1.8V regulator
- Nízká spotřeba: n**Hold, Sniff, Park, Deep sleep Mode**
- Napájení od 3.0v do 3.6v
- Podpora až 7 zařízení: **SCO, ACL, Piconet<7>**
- Interface: USB, SPI s UART&PCM
- HCI nebo SPP firmware k dispozici
- rozměry 25 x 14.5 x 2.2 mm¨

Modul je možné použít do mnoha zařízení jako je: Notebook PC, PDA, Cordless headset, Digital camera & printer, GPS, POS, Barcode Reader, Domestic and industrial applications

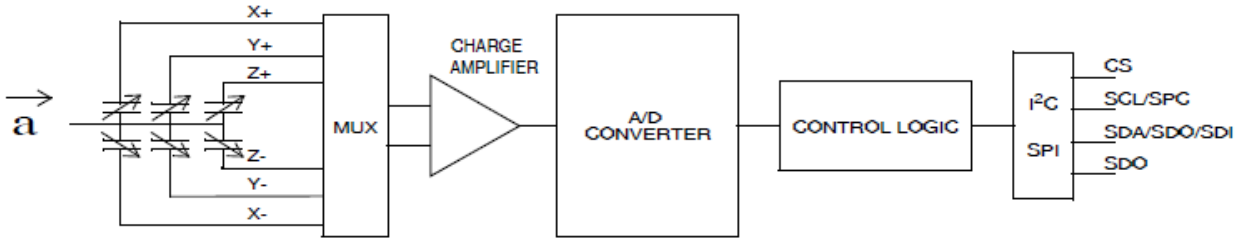
10 Akcelerometr KAmodMEMS2

KAmodMEMS2 je tříosý akcelerometr ze stavebnice KAmod od firmy KAMAMI, vybavený senzorem LIS35DE. Jedná se o kapacitní senzor zrychlení, pracující jako proměnný kondenzátor. Jedna elektroda kondenzátoru je pevná, druhá je pohyblivá. Pokud na pohyblivou elektrodu působí zrychlení, začne se přibližovat resp. oddalovat od druhé elektrody a dojde ke změně kapacity. [8]



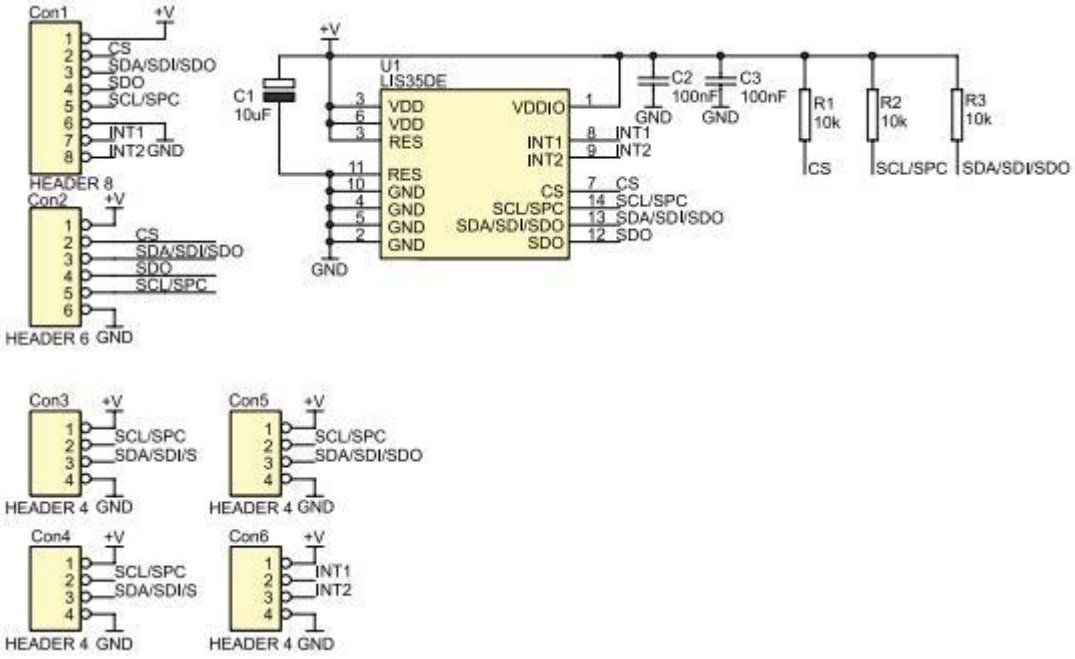
Obrázek 11 - KAmodMEMS2

Princip vyhodnocení je znázorněn na následujícím obrázku (12). Kapacity jednotlivých os jsou převedeny blokem CHARGE AMPLIFIRE na napětí. Toto napětí je pomocí A/D převodníku dekódováno do osmibitového registru. Přístup k těmto registrům resp. komunikaci zajišťuje blok CONTROL LOGIC. [8]



Obrázek 12 – KAmodMEMS2 princip vyhodnoceni

Pro komunikaci je modul vybaven digitálními rozhraními SPI a I2C. Na přípravku je hned několik paralelních výstupů. Pro I2C je přípravek konektory Con3, Con4, Con5. Con4 a Con5 jsou pro standartní I2C piny (kompatibilní s CAB_HU04 kabelem, Zl15AVR atd.) Con6 se používá pro připojení ke generátoru přerušení pro I2C. Pro SPI je to Con2 (kompatibilní s CAB_HU06 kabelem). Na Con1 je také SPI linka rozšířena o generátory přerušení (INT1, INT2).

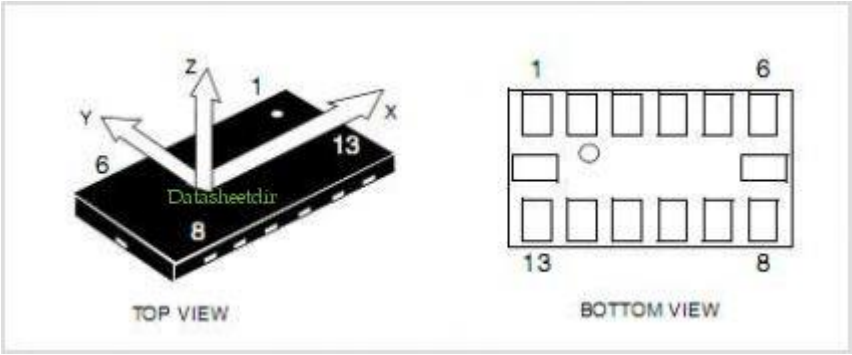


Obrázek 13 - Schéma konektorů

Vlastní spotřeba senzoru dosahuje velmi ekonomické hodnoty do 1mW. Senzor může pracovat s napájecím napětím v rozmezí 2,16V až 3,6V. Bylo zvoleno běžně používané napájecí napětí 3,3V.

LIS35DE nabízí uživateli dva citlivostní rozsahy. Je možné zvolit hodnotu ±2G nebo ±8G. Senzor je vybaven několika vestavenými frekvenčními propustmi a umožnuje dále několik nadstandardních funkcí jako je detekce kliknutí, popř. dvojkliku.

Na obrázku je vidět LIS35DE a orientace os vzhledem k akcelerometru. Na přípravku KAmodMEMS2 jsou rovněž znázorněny osy.



Obrázek 14 - LIS35DE

11 Realizace úlohy

Pro vytvoření ukázkové úlohy byly využity výše popsané zařízení. Pro systém byl zvolen FreeRTOS, který se stará o periodické spouštění úloh. Program byl vyvíjen ve vývojovém prostředí LPCXpresso v7, kde pomocí USB konektoru a vestavěném JTAGu na desce byl program jednoduše testován.

K vývojové desce LPC1769 byl připojen akcelometr KAmodMEMS2 pomocí I2C komunikace, kde LPC1769 byl nastaven jako master a akcelometr jako slave s adresací 0011101xb. Bit x je nejnižší bit, který slouží pro značení zápisu (0) nebo čtení (1). Zde nastal první problém. Na ovládání a nastavení I2C komunikace nebyla nalezena žádná dokumentace. Od výrobce vývojové desky existuje pouze demonstrační úloha, která pro začátečníky či průměrného uživatele není nijak dobře popsána. Z nedostatku informací bylo na základě předchozích zkušenostech předpokládáno, že do struktury pro řízení komunikace se do proměnné „xfer.slaveAddr“ uloží celá adresa zařízení (akcelerometru). Tuto adresu si softwarová komunikace sama upravuje, resp. upravuje nejnižší R/W bit, jako to je například v mikrokontrolérech PICAXE. Ovšem v případě LPC se do proměnné uloží pouze 7 bitů bez nejnižšího bitu. Při komunikaci se pak tato hodnota v proměnné odrotuje doleva a přidá se poslední bit signalizující čtení či zápis. V případě, kdy se vložila celá 8-bitová adresa nenastala žádná „chyba“ při běhu programu. Adresa se odrotovala o jeden bit doleva, přidal se poslední bit a vznikla úplně jiná adresa. Komunikace pak vracela status NAK (negative-acknowledge) tj. „negativní potvrzení,“ protože na lince nebylo žádné zařízení s takovou adresací. Se správnou adresací akcelerometru byla pak komunikace napájena napětím 2.5V s 3K0 pull-up rezistory. Na následujícím obrázku je část kódu v jazyce C, který se stará o inicializaci I2C komunikace v LPC i v akcelerometru.

```
233
234     i2c_app_init(I2C0, SPEED_400KHZ);
235     slaveAddress = 0x1D;
236     slaveAddress &= 0xFF;
237     xfer.slaveAddr = slaveAddress;
238     xfer.txSz = 2;
239     buffer[0][0] = 32;
240     buffer[0][1] = 199;
241     xfer.txBuff = buffer[0];
242     tmp = Chip_I2C_MasterSend(i2cDev, xfer.slaveAddr, xfer.txBuff, xfer.txSz);
243
```

Obrázek 15 – Inicializace I2C v LPC1769

Na řádku 234 je funkce, která nastaví I2C v LPC1769 pro přednastavené piny I2C0 s rychlostí 400KHz. Oba parametry jsou makra, přednastavená v hlavičkovém souboru. Na dalších řádcích (235-237) se uloží 7-bitová adresa akcelerometru do proměnné slaveAddr, která je ve struktuře xfer. Na dalším řádku se definuje počet odchozích bitů pro následující komunikaci. Do předem vytvořeného bufferu se vloží data, které se budou při následující komunikaci odesílat. Pointer na tento buffer se předá struktuře zajišťující komunikaci na řádku 241. Na posledním řádku se zavolá metoda, která zahájí komunikaci a odešle všechny data. Tato metoda vrací počet úspěšně přenesených bytů. Po odeslání adresy akcelerometru je první byte (hodnota 32) adresa registru sloužící pro základní nastavení akcelerometru. Další byte jsou nastavení. Podle datasheetu senzoru byl registr nastaven takto :11000111. Bity zleva značí: 400Hz data rate (1), zapnutí senzoru (1), citlivost +-2g (0), následující 2 bity musí být 0 a poslední 3 bity slouží pro zapnutí jednotlivých os v pořadí x-y-z.

Podobným způsobem probíhá i přenos dat z jednotlivých os akcelerometru. Na následujícím obrázku je opět část kódu z projektu. Kód řeší načítání dat ze senzoru do bufferu.

```
256
257         xfer.txSz = 1;
258         buffer[0][0] = (uint8_t)169;
259         xfer.txBuff = buffer[0];
260         xfer.rxSz = 5;
261         xfer.rxBuff = buffer[1];
262         I2C_STATUS_T value = Chip_I2C_MasterTransfer(i2cDev, &xfer);
263         if (value != 0)printf("I2C status :%d", (int)value);
264
```

Obrázek 16 – Načítání dat ze senzoru

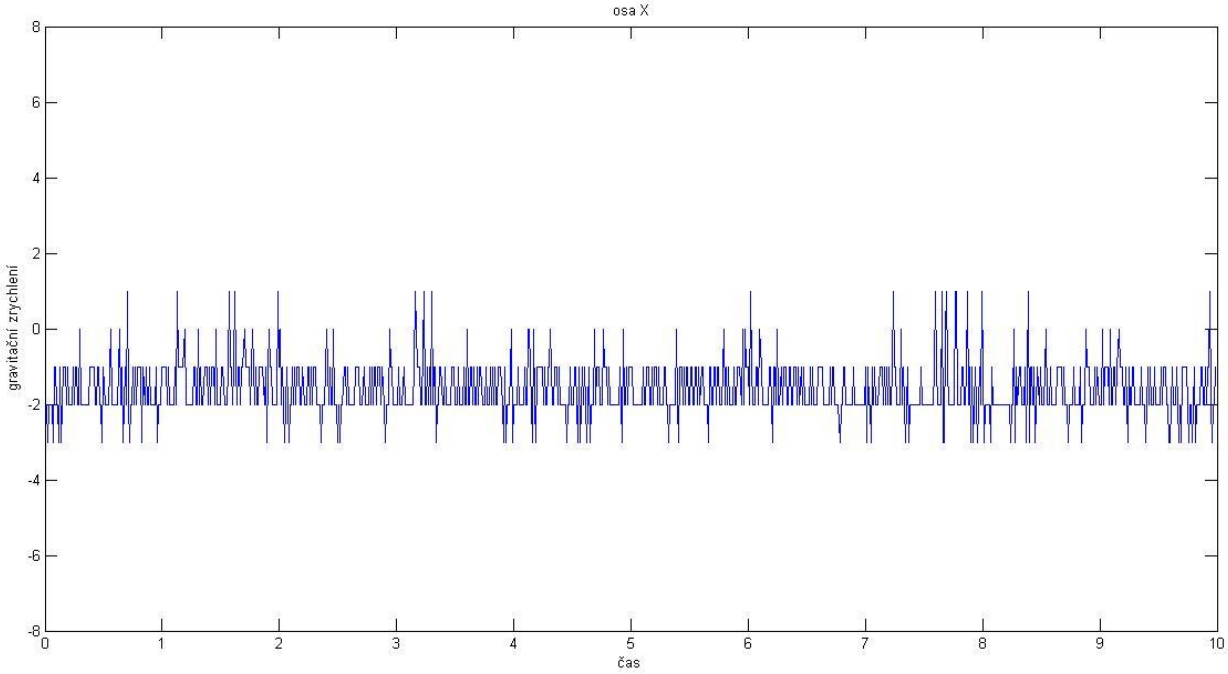
Tentokrát je poslán do akcelerometru pouze jeden byte (10101001), obsahující adresu registru osy X. Kde nejvyšší bit značí auto-increment. To znamená, že se bude chtít načítat z akcelerometru několik dat najedou. První data jsou z poslané adresy registru (00101001) a další jsou vždy z registru o 1 větší, než data z předchozího registru. Přenos dalších byte ukončí master. Protože jsou adresy os v akcelerometru uloženy ob jeden byte, je třeba načíst 5 bajtů. Do rxSz tedy uložíme 5 a do rxBuff uložíme poiter na buffer pro ukládání načtených dat. Na řádku 262 se volá funkce, který opět zahájí přenos. Tahle ale nejdřív odešle data a poté přijme příchozí. Funkce vrací stav po ukončení nebo přerušení komunikace. Hodnota 0 značí, že vše proběhlo v pořádku. V opačném případě vrátí číslo reprezentující danou chybu (1=NAK, 4=BUSY atd.).

K LPC1769 byl dále připojen bluetooth modul BTM-110. Pro komunikaci byl zvolen UART. Zde byl další problém. K bluetooth modulu bylo nalezeno pouze pár důležitých informací. V datasheetu je uvedeno pár základních informací (např. napětí) a schéma s popsanými piny. Dále se podařilo najít defaultní nastavení modulu včetně hesla pro spárování. S těmito informacemi se podařilo zprovoznit komunikaci s LPC1769 přes UART s rychlostí 19200 baudů, délkou slova 8 bitů, 1 stop bit a žádnou kontrolou parity. Koncové zařízení (pc, chytrý telefon) se pak spáruje s modulem a po otevření terminálu s nastavením komunikace jako je v modulu lze přijímat data. LPC1769 pošle data po bytu do bluetooth modulu, který pak přepošle byty bezdrátově do koncového zařízení.

Program nejprve načítá data ze vše os v akcelerometru s periodou 10ms. Jelikož má vývojová deska dostatek místa na ukládání naměřených hodnot ve stanoveném časovém intervalu 10 sekund, není třeba připojit externí paměť. Po navzorkování 10 sekundového pohybu, jsou data poslána do PC ve tvaru x = [x_1, x_2, x_3, …x_n]; (platí i pro y a z). Výsledné data se pak v pc vložily do grafu.

12 Výsledná data

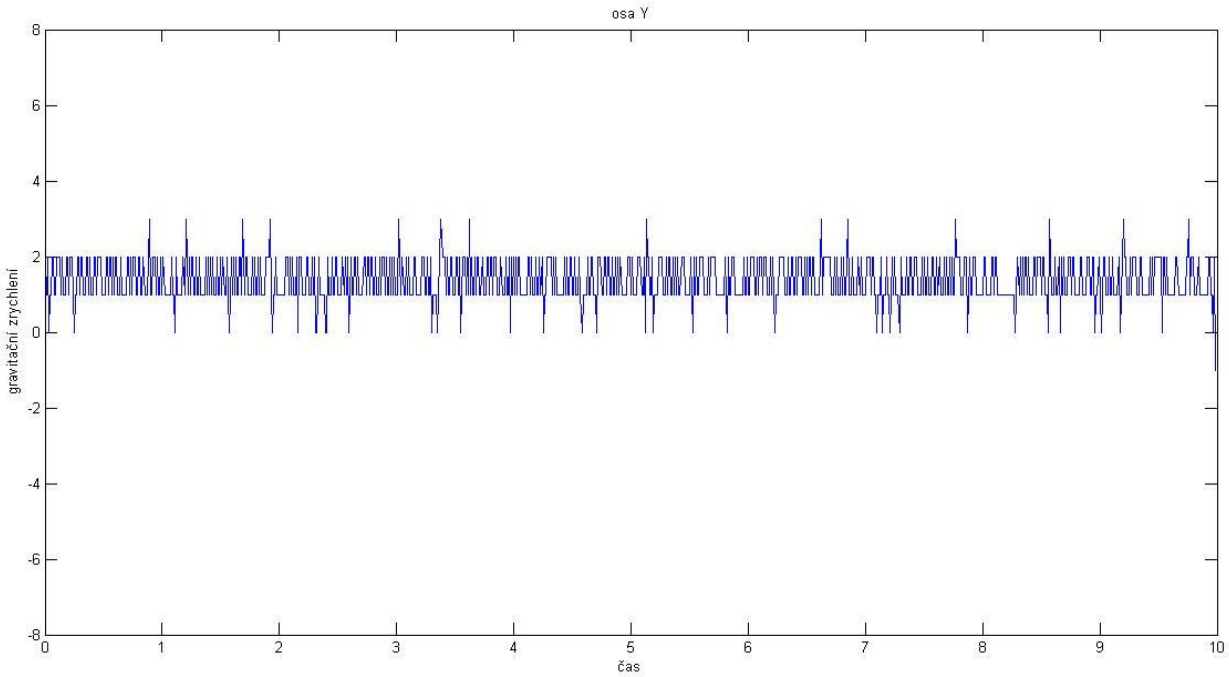
Nejprve byl navzorkován klidový stav. Akcelerometr byl položen na vodorovnou podložku. Na následujícím grafu je znázorněno gravitační zrychlení z osy x.



Obrázek 17 – Klidový stav osa X

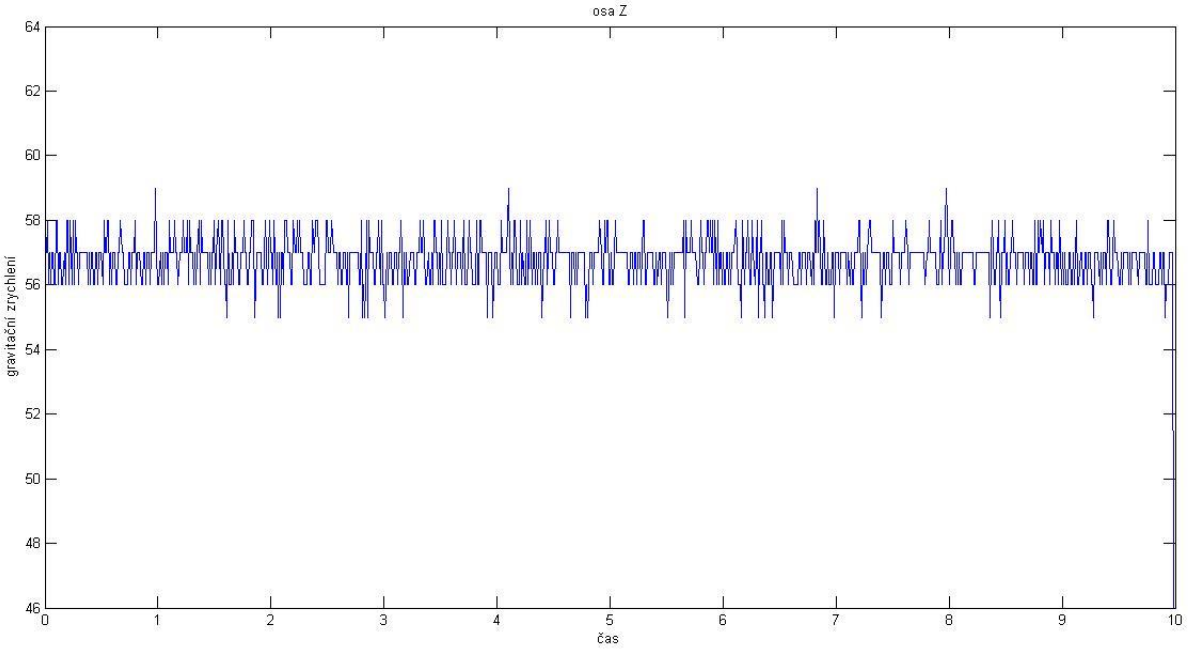
Data z os jsou velké 1 byte tj. 256 hodnot. Citlivost byla nastavena na +-2g. Z toho lze říci, že 0 – 2g odpovídá hodnotám 0-127, pro -2 – 0g odpovídá -128 – 0. Z grafu je vidět šum senzoru v podobě oscilace od -3 do 1.

Na dalším grafu je klidový stav osy Y. Zde je vidět stejně veliký šum od 0 do 3.



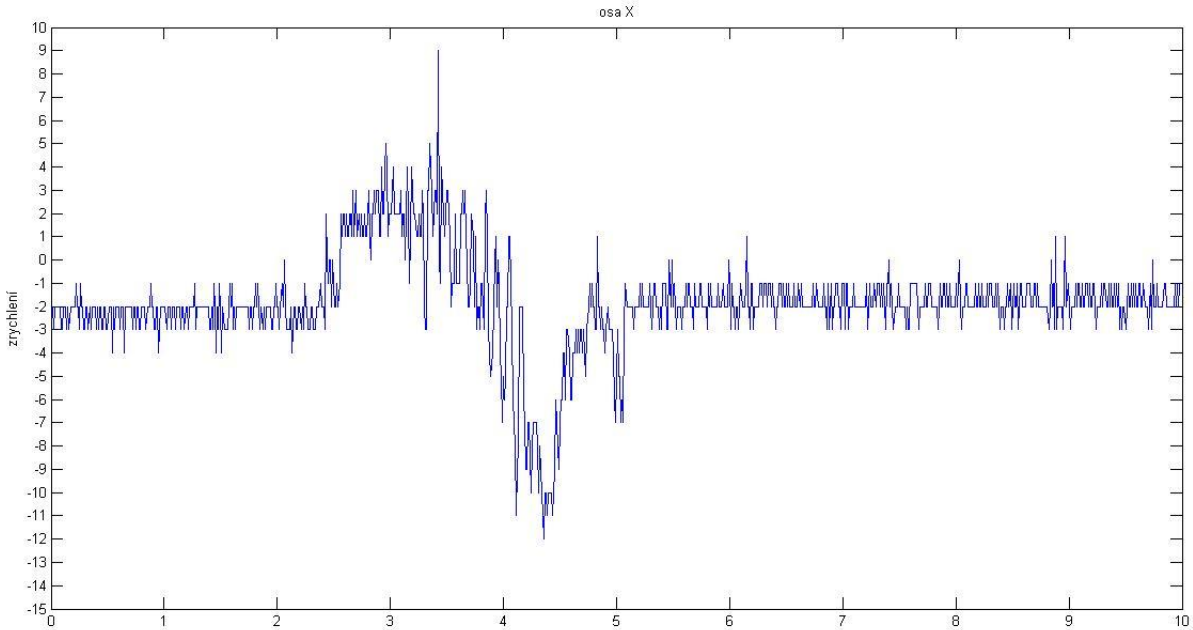
Obrázek 18 – Klidový stav osy Y

Třetí graf znázorňuje osu Z. Zde se projevilo tíhové zrychlení Země. Protože směr osy je stejný jako směr tíhového zrychlení, hodnoty tedy ukazují 1g tj. 9,81 m/s². Opět je v grafu vidět šum. Na ose Z bude zrychlení 1g odpovídat hodnotě 57.



Obrázek 19 – Klidový stav osy Z

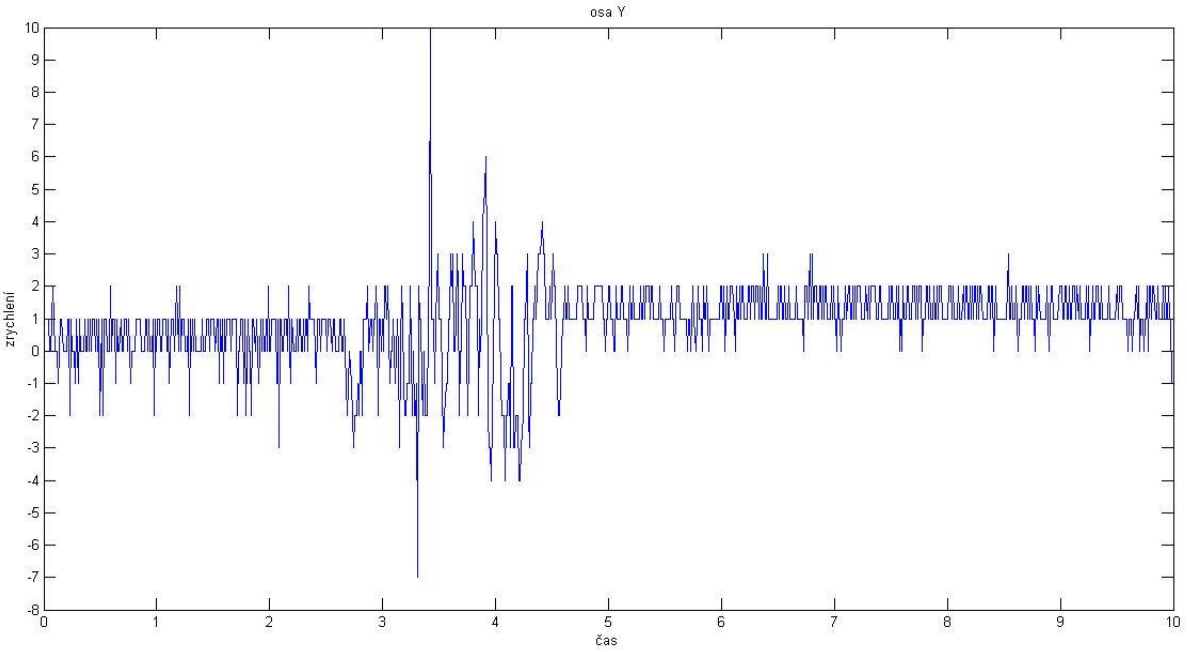
Další zkoumaný pohyb je přímočarý pohyb ve směru osy X. Osa Y byla ve vodorovné poloze a osa Z směrovala do středu Země. Senzor při pohybu urazil vzdálenost 101cm. Průběh pohybu je znázorněn na následujícím grafu osy X.



Obrázek 20 – Přímočarý pohyb osa X

Zde je vidět, že senzor stál asi 2,5 vteřiny a poté se začal rozjíždět. Zrychlení se nejdříve zvětšovalo a zařízení se rozjíždělo stále rychleji. Potom se zrychlení zmenšovalo až na konstantní rychlost (cca v čase 3,9 vteřiny). Následně začalo zařízení brzdit, to je znázorněno v záporných číslech. Opět je zde vidět rostoucí intenzita brždění následována snižující se intenzitou brždění do téměř nulové rychlosti. Nulová rychlost nastala v čase 5, kde je vidět rychlý nárůst brždění (zrychlení v opačném směru). To mohlo být způsobeno třecí sílou, která byla již dostatečně velká, aby zařízení „zasekla“ na místě.

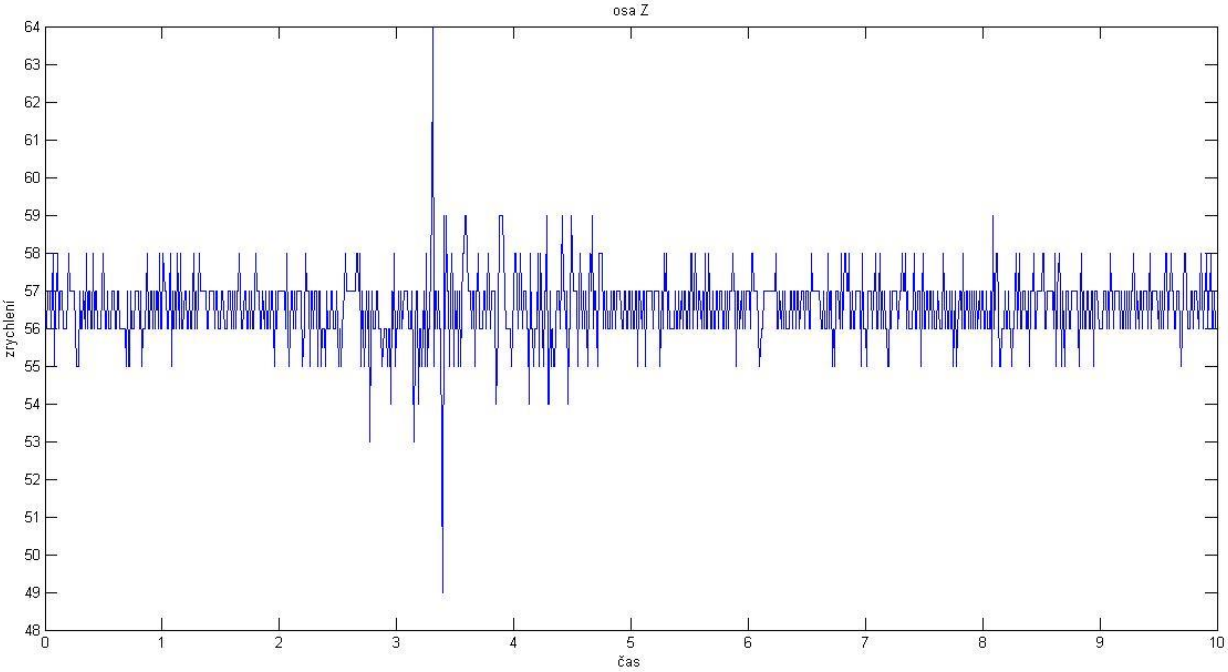
Graf osy Y by měl zůstat v klidovém stavu, protože se jednalo přímočarý pohyb a zařízení tedy nijak nezatáčelo ani se nenaklánělo.



Obrázek 21 Přímočarý pohyb osa Y

Kromě šumu jsou v grafu vidět i jiné hodnoty. Ty mohly být způsobeny vibracemi vlivem nerovného povrchu.

Osa Z by měla ukazovat po celou dobu pohybu pouze tíhové zrychlení, protože senzor nijak nestoupal ani neklesal (nejel ani do kopce ani z kopce).



Obrázek 22 – Přímočarý pohyb osa Z

Jak je vidět na grafu, osa Z ukazuje tíhové zrychlení. Ovšem v čase 3,25 ukazuje velkou odchylku od celého průběhu. Pokud se podíváme i na graf osy Y, je vidět, že odchylka je ve stejnou dobu. Z toho je patrné, že se jednalo o otřes, způsobený nerovností povrchu.

12.1 Výpočet rychlosti a dráhy

Z naměřených hodnot (zrychlení) lze nepřímo určit překonanou vzdálenost a okamžitou polohu. Pokud je předmět v klidu nebo v rovnoměrném přímočarém pohybu, je zrychlení nulové. Při nabírání rychlosti je zrychlení kladné, při zastavování záporné. Z této veličiny lze určit okamžitou rychlost a z rychlosti překonanou vzdálenost. Výpočet těchto veličin je zjištěn pomocí numerické integrace. Obecný zápis integrace pro určení vektoru okamžité rychlosti v čase t pro jednu osu je:

$$v(t) = \int_{t_0} a(t) * dt + v_0,$$

Kde v_0 je rychlost v čase počátku integrace t_0, a je zrychlení v časovém okamžiku t. Dvojitou integrací signálu z akcelerometru lze určit překonanou vzdálenost x, kterou urazil určitý bod za danou dobu. Pro přírůstek dráhy platí vztah:

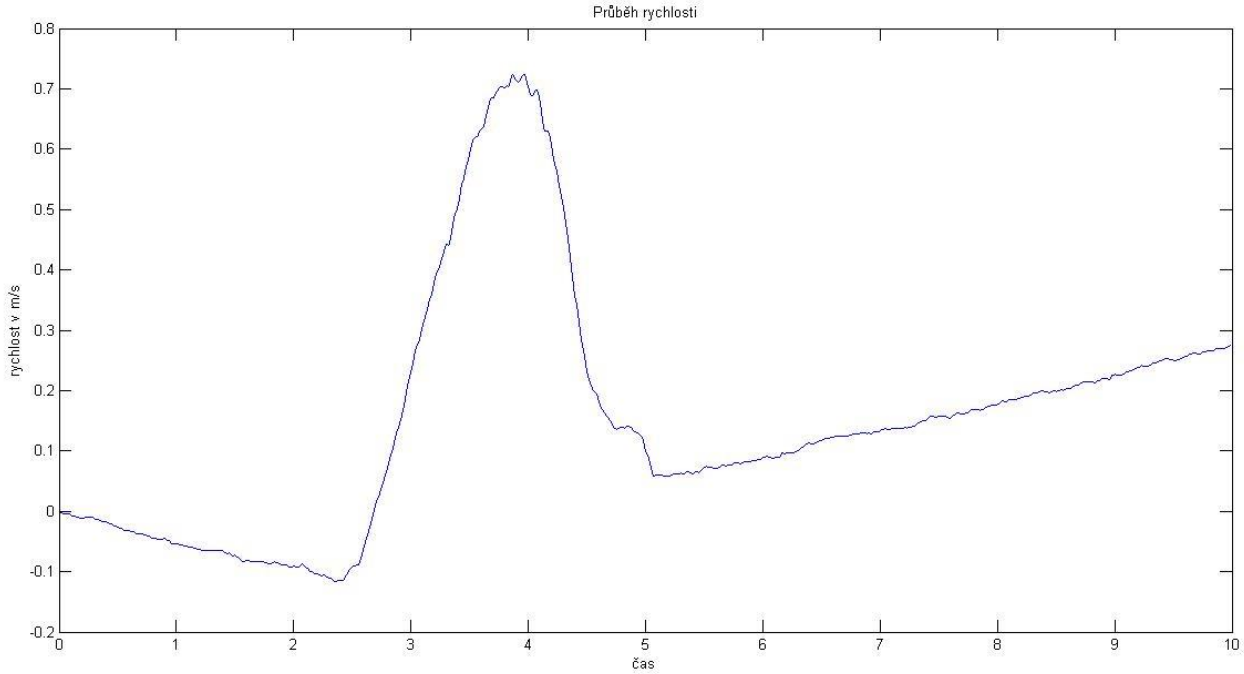
$$x(t) = \int_{t_0} v(t) * dt + x_0 = \iint_{t_0} a(t) * dt * dt + v_0 * t + x_0,$$

Kde v je rychlost v časovém okamžiku t, x_0 je pozice v čase počátku integrace t_0 a ostatní parametry jsou stejné jako u předchozího vzorce. [7]

Na výpočet bylo použit přesnější numerická derivace zvaná Lichoběžníková metoda.

$$v_i = \frac{(a_i + a_{i-1})}{2} * \Delta t + v_{i-1},$$

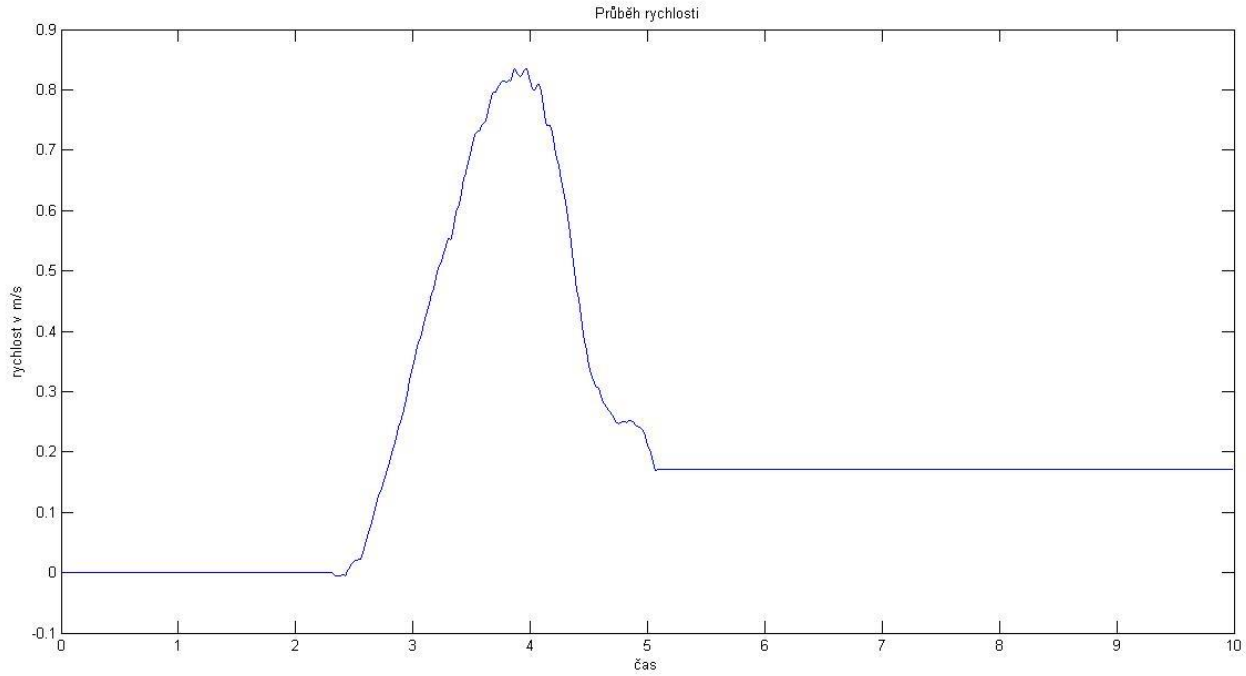
S touto metodou byl spočítán průběh rychlosti. Protože senzor ve vodorovné poloze ukazuje hodnoty okolo čísla -2, byly hodnoty kvůli přesnějšímu výpočtu posunuty o hodnotu +2. Průběh rychlosti v závislosti na čase je znázorněn na následujícím grafu.



Obrázek 23 – Průběh rychlosti se šumem

V grafu je vidět, že díky šumu je výpočet zkreslen. Graf ukazuje konstantní nárůst rychlosti v opačném směru před samotným pohybem senzoru. Pak je vidět rostoucí rychlost až na maximální rychlost 0,7239 m/s v čase 4. V čase 5 by se mělo zařízení zastavit a rychlost by

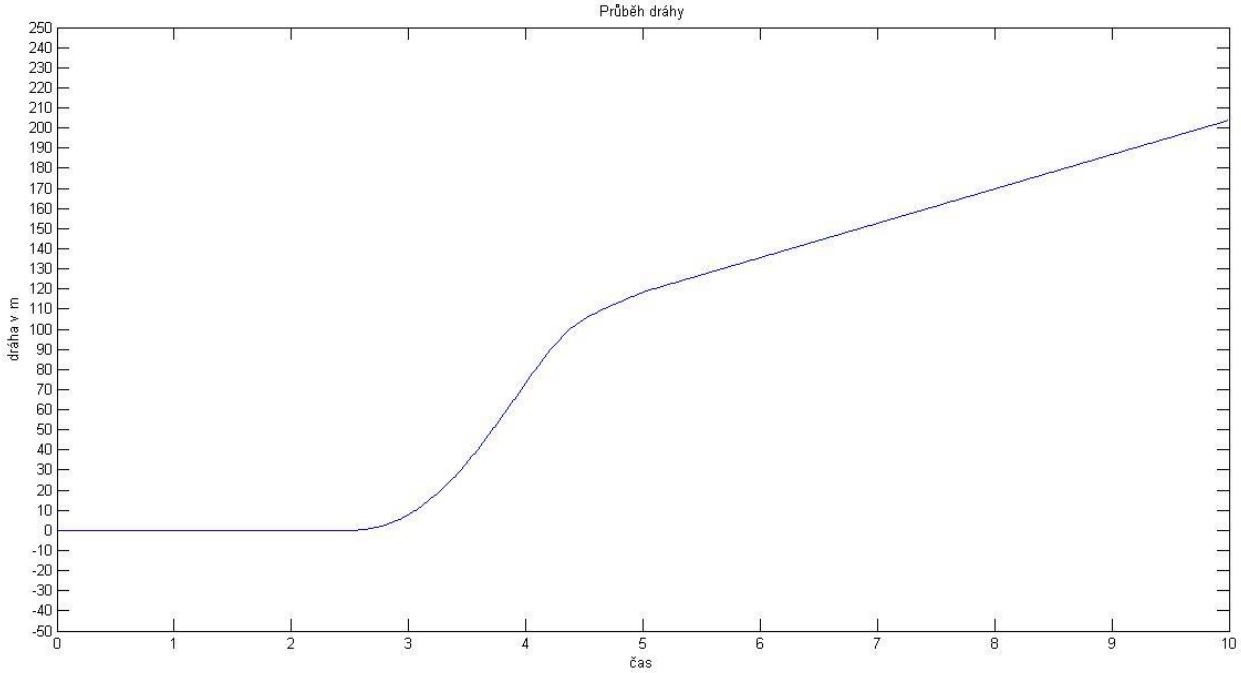
měla být nulová. Ovšem zde malá rychlost zůstala a díky šumu se začala zvětšovat. Proto byl odstraněn šum před a po samotném pohybu. Výsledek by měl být méně zkreslen.



Obrázek 24 – Průběh rychlosti bez šumu

Zde už zůstává rychlost nulová do začátku pohybu. Maximální rychlost je díky nulové rychlosti na startu vyšší: 0,8348 m/s. Po zastavení senzoru zůstala spočítaná rychlost nenulová: 0,1775 m/s. To mohlo být způsobeno, jak šumem v průběhu pohybu, tak malým rozsahem hodnot citlivosti. Největší citlivost senzoru LIS35DE, jak už bylo řečeno, lze nastavit na +-2g. Celkový rozsah 4g je rozložen do 256 hodnot. Pro tento přímočarý pohyb bylo potřeba jen 22 hodnot. Pokud by byl pohyb výraznější (větší rychlení), nebo by byla natavena větší citlivost, byl by výsledek přesnější. Z rychlosti byla spočítána i dráha. Její průběh lze pozorovat na grafu na následující stránce.

Z grafu je patrné, že dráha v počátku startu pohybu začala „exponenciálně“ narůstat až do bodu, kdy zařízení začalo brzdit. Zde by se graf měl stále více přibližovat konstantní funkci. Díky nenulové spočítané rychlosti na konci pohybu dráha lineárně narůstá místo toho, aby byla konstantní. V čase 5, kdy se pohyb zařízení zastavil, je překonaná vzdálenost 118 cm. To je o 17% více, než je reálně ujetá vzdálenost.



Obrázek 25 – Průběh dráhy

13 Závěr

Úkolem této bakalářské práce bylo seznámení se s vývojovými deskami LPCExpresso od firmy NXP, které jsou osazeny mikrokontroléry řady LPC1xxx. Zaměřit se na komunikaci s dalšími obvody a ovládání digitálních vstupů a výstupů a ověřit jejich funkčnost. Dále navrhnout ukázkovou úlohu, která bude demonstrovat využití mikrokontroléru v mobilním robotu. Navrženou aplikaci realizovat a zhodnotit zjištěné výhody a nevýhody implementace dané vývojové desky.

V práci lze pokračovat přidáním dalších zařízení připojitelných přes I2C komunikaci. Díky velkému výpočetnímu výkonu lze do programu implementovat další úlohy reagující na různé vstupy. Také lze pomocí bluetooth mikrokontrolér ovládat.

V bakalářské práci jsou jednotlivá zařízení popsána a vysvětlena jejich funkčnost. Jsou zde také vysvětleny protokoly, díky kterým mikrokontrolér komunikuje s periferními zařízeními. Dále byla vytvořena aplikace, demonstrující využití mikrokontroléru a připojených periférií jako je MEMS senzor a bluetooth modul v praxi. Úloha snímá pohyb ve formě zrychlení daného objektu v časovém rozmezí 10 sekund a poté naměřené údaje posílá bezdrátově do počítače či mobilního zařízení. Program je napsán v jazyce C a pro operační systém FreeRTOS. Načítá data ze všech 3 os z akcelerometru s periodou 10ms a ty ukládá do vnitřní paměti vývojové desky LPC1769. Po navzorkování průběhu pohybu v čase 10 sekund jsou všechny data os x, y a z poslány do bluetooth modulu, který je pak přepošle spárovanému a připojenému zařízení. Data jsou pak vyhodnocena a znázorněna v grafech.

Závěrem této práce lze říci, že se podařilo navrženou aplikaci realizovat a získat výsledky z naměřených dat z akcelerometru. Vypočítaná uražená vzdálenost objektu byla v porovnání s reálnou uraženou vzdáleností o 17% větší. Nepřesnost výsledků mohla být způsobena, jak šumem senzoru, který je u těchto typů senzorů problém, tak malým rozsahem hodnot popisující pohyb. Odstup signálu od šumu byl v tomto měření malý. Výsledky z takových měření jsou spíše orientační. Pro přesnější výsledky je třeba zvolit citlivější zařízení pro pohyb s menším zrychlením.

Seznam použité literatury

[2] HRBÁČ, Jan. Aplikace počítačů v měřících systémech pro chemiky. Olomouc, 2012. Univerzita Palackého, Přírodovědecká Fakulta, Katedra Fyzikální chemie.

[3] Bc. SEDLÁŘ, Petr Digitálně řízený rezistor. Brno 2009. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav radioelektroniky. Vedoucí práce Ing. Roman Šotner.

[4] OLEJÁR, Martin Stručný popis sběrnice I2C a její praktické využití k připojení externí eeprom 24LC256 k mikrokontroléru PIC16F877[online]. HW.cz URL: <http://www.hw.cz/navrh-obvodu/strucny-popis-sbernice-i2c-a-jeji-prakticke-vyuziti-k-pripojeni-externi-eeprom-24lc256>

[5] BARRY, Richard. Using the FreeRTOS Real Time Kernel. 1. vyd. London: Real Time Engineers Ltd., 2010. ISBN 978-1-4461-7030-4.

[6] HORÁK, M. Bezdrátový mesh protokol pro FreeRTOS. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2014. 36 s. Vedoucí bakalářské práce Ing. Vladimír Červenka.

[7] Ing. KUTÍLEK, Patrik, Ph.D. Ing. ŽIŽKA, Adam. Úloha KA03/č. 4: Měření kinematiky a dynamiky pohybu končetin pomocí akcelerometru. České vysoké učení technické v Praze, Fakulta biomedicínského inženýrství. 11.10.2010 – 28.2.2013.

[8] ŘEHOŘEK, Adam Popis periferií připojitelných k mikrokontroléru. Liberec 2014, Bakalářský projekt. Technická univerzita v Liberci, Fakulta mechatroniky, informatiky a mezioborových studií. Vedoucí bakalárské práce Ing. Miroslav Holada, Ph.D.